**Análisis del *stack* propuesto en Stack\_1.docx**

| **Capa** | **Tecnologías actuales** | **Valor que aportan** | **Observaciones** |
| --- | --- | --- | --- |
| **Frontend** | Vue 3 + Tailwind + Vite | Micro‑frontends, DX rápida | Correcto, pero la demo MVP la lleva React; conviene unificar o justificar la convivencia para evitar duplicar conocimiento. |
| **Backend API** | FastAPI + Uvicorn | Rendimiento async, OpenAPI nativo | Excelente elección; añadir **Pydantic v2** y patrón *service‑layer* para escalar código. |
| **Tokens** | Docker Alpine + gVisor | Imágenes ligeras & *sandboxing* | Añadir **distroless** en tokens estáticos y activar *seccomp/apparmor* por defecto. |
| **Orquestación** | K3s/Helm | Minimal K8s, ideal para demo | Mantener K3s para laboratorio y Terraform‑EKS/GKE para producción. |
| **Mensajería** | RabbitMQ | Fiabilidad y *routing* | Incluir **Dead‑Letter Exchange** para reintentos y métricas. |
| **BBDD** | PostgreSQL + TimescaleDB | OLTP + series temporales | Separar pods y discos; habilitar **pgAudit** y **WAL‑G** para backups continuos. |
| **Seguridad** | Vault + OPA | Secrets + política | Añadir autenticación **mTLS** entre pods y escaneo Trivy en pipeline. |
| **Observabilidad** | Prometheus + Grafana Loki | Métricas + logs | Incluir **Tempo** para *tracing* y alertas SLO con **Alertmanager**. |

**Mejoras recomendadas**

1. **Unificación Frontend**
   * ➜ Migrar la demo React a **Vue 3** o viceversa para no mantener dos stack de UI.
   * Si se decide convivir, aislar cada micro‑frontend con **Module Federation**.
2. **Seguridad *Zero‑Trust***
   * Iniciar sesión mutua con **SPIFFE/SPIRE** entre tokens y API.
   * Políticas OPA que validen el manifest.yml antes de lanzar un token.
3. **Gestión de costes y escalado**
   * Activar **Karpenter** (AWS) o Cluster Autoscaler con *spot nodes*.
   * Medir coste por token (CPU / RAM × tiempo) y exponerlo en Dashboard.
4. **Ciclo CI/CD endurecido**
   * Escaneo **Trivy + Grype** en cada docker/build-push-action.
   * **Cosign**: firma y verificación de todas las imágenes en el registry privado.
5. **Observabilidad avanzada**
   * Trazas distribuidas con **OpenTelemetry + Tempo**.
   * Dashboards pre‑hechos: *Latency per Token*, *Errores WhatsApp*, *Coste/hora*.
6. **Test de resiliencia**
   * Incluir **ChaosMesh** (K8s) o **Litmus** para matar pods de tokens y validar autorecuperación.
7. **Programa de *On‑boarding* de partners**
   * Publicar plantilla Cookiecutter que genere esqueleto de token con pruebas y manifiesto.
   * Pipeline de validación automática que verifique licencias, consumo y seguridad.
8. **Compliance‑ready**
   * Mapear logs a **MITRE ATT&CK** y preparar exportación a SIEM externo vía **OTel exporter**.
   * Plantilla de informe GDPR/NIS2 con evidencias automáticas (pgAudit, Vault audits).

**Acciones de corto plazo (para el MVP de 2 meses)**

| **Prioridad** | **Tarea** | **Responsable sugerido** | **Entregable** |
| --- | --- | --- | --- |
| **Alta** | Integrar escaneo Trivy y firma Cosign en GitHub Actions | DevOps | Pipeline verde con firma verificable |
| **Alta** | Unificar UI o documentar estrategia de micro‑frontends | Líder técnico | ADR (Architecture Decision Record) |
| **Media** | Desplegar Vault dev‑mode + secret injection vía CSI | DevOps | Vault Helm chart configurado |
| **Media** | Panel Prometheus + Loki básicos + alertas demo | QA/Observabilidad | Dashboard JSON exportable |
| **Baja** | Prototipo OPA policy para token manifest.yml | Back‑end Dev | Repositorio de políticas |

**Checklist extendido para la nueva capa de mejoras**

**🔒 Seguridad & Compliance**

* mTLS entre pods emisor‑receptor validado con SPIRE.
* Trivy ejecuta --severity CRITICAL,HIGH.
* Regla OPA: prohibir imágenes latest en producción.
* Vault audit → Loki con etiqueta vault\_audit=true.

**📊 Observabilidad**

* Exporter OpenTelemetry activo en API y tokens.
* Dashboard “Coste por token” con datos Prometheus (CPU × precio nodo).
* Alertmanager: SLO “99 % jobs success” con notificación a Slack.

**⚙️ Build & Deploy**

* Cosign keyless signing vía GitHub OIDC.
* GitHub Environment Production con *required approvals* ≥ 1.
* Helm chart lint con ct lint.

**💸 Cost‑control**

* Karpenter/CA escala a 0 cuando no hay jobs.
* Etiqueta team=sentinelcore en todos los nodos → Cost Explorer.
* Informe semanal CSV coste/token.

Con estas mejoras el *stack* PlatSentinel queda **alineado con prácticas DevSecOps de 2025**, preparado para demostraciones robustas y para escalar a producción regulada cuando el MVP madure.